ZebraSound : architecture and notes

1 Introduction : notation and base principle

The sonification is based on the mean fluorescence signals computed over a set or ROI. An ROI (or neuron) is represented by an FM operator.

Mapping :

- Fluorescence : Amplitude
- Group parameter : Modulation index
- First derivative : harmonicity (not used, set to 1)

1.1 Notation

We note $\mathbf{F}(n) = [f_0(n), ...f_i(n), ..., f_{N_{ROI}-1}(n)]^T$ the vector containing the mean fluorescence in each ROI at the current frame n. The element $f_i(n)$ designates the fluorescence associated with the i^{th} ROI.

We note $\mathbf{G}(n) = [g_0(n), ...g_j(n), ..., g_{N_G-1}(n)]^T$ the *group vector* which contains the parameters computed for each group j, $0 < j < N_G$, with N_G the number of groups (nine in our case).

Therefore, $\mathbf{G}(n)$ can be computed with a matrix operation : $\mathbf{G}(n) = \mathbf{A}.\mathbf{F}(n)$, with $\mathbf{A}$ a matrix of dimension $N_G \times N_{ROI}$. The coefficients of $\mathbf{A}$ then determine which parameter is being computed. We note that $\mathbf{G}(n)$ can be computed using the first time derivative or $\mathbf{F}(n)$.

Four parameter are currently implemented :

- Mean fluorescence in the group
- ratio of activated neurons
- first derivative of the above

These parameters can be computed either on the fluorescence or on its first derivative.

Due to the high dimensions of our data, we prefer direct computation of $\mathbf{G}(n)$ over a costfull matrix operation.

The raw data goes through a preprocessing and a normalization. The resulting signal vectors are noted $\mathbf{F_p}(n)$ and $\mathbf{F_n}(n)$ respectively.

The scaled signals are notes with the subscript s, the driving signals are noted with the subscript d.

1.2 overall structure

The overall structure is represented figure 1.

Amplitude driving The raw data go through the processing, which consists in a preprocessing and a normalization. Both are discussed in section 2. The normalized signal and its first derivative (backward finite differences, $o(h^2)$. For each frame the current vector fn is obtained from the processed data to be used in the sonification.

The signal $\mathbf{F_n}(n)$ is threshed and scaled. $\frac{\mathrm{d}\mathbf{F_n}(n)}{\mathrm{d}t}$ is scaled and used to sharpen $\mathbf{F_s}(n)$ according to the expression

$$\mathbf{F_d}(n) = (1-a)\mathbf{F_s}(n) + a\left[\frac{\mathrm{d}\mathbf{F_n}(n)}{\mathrm{d}t}\right]_s \tag{1}$$

Where a is an arbitrary coefficient, and $[.]_s$ designates the scaling operation.

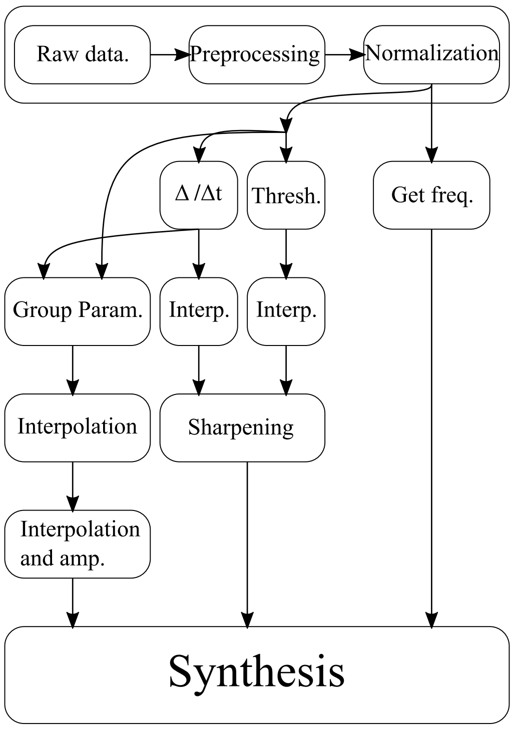

Figure 1: overview

Modulation index driving The modulation is driven by the scaled group parameter multiplied by an arbitrary coefficient k , so that$\mathbf{G_d}(n) = k\mathbf{G_s}(n)$. The amplification by a factor k is used to set the range of variation of the modulation index. For each frame the values are routed and the parameter is computed. The computation is detailed in section 3.2.

Frequency The carrier frequency of each operator is obtained by referencing the cluster the neuron belongs to. The note associated with each group is set by the user.

2 Data processing

In this section we detail the computation of the processed and the normalized signals.

2.1 Preprocessing

Two preprocessing methods are implemented, the *average* (eq. 2) and the *standard deviation* (eq. 3).

$$\mathbf{F_p}(n) = \frac{|\mathbf{F}(n) - f_{mean}|}{f_{mean}} \tag{2}$$

$$\mathbf{F_p}(n) = \frac{|\mathbf{F}(n) - \sigma^2|}{f_{mean}} \tag{3}$$

f_{mean} designates the vector of size N_{ROI} which contains the mean fluorescence in each ROI, over the whole recording. Therefore it is expressed as

$$f_{mean} = \left[mean(f_0), ..., mean(f_i), ..., mean(f_{N_{ROI}-1)}\right]^T \tag{4}$$

2.2 Normalization

In the case of the data obtained from FIJI, the fluorescence values are between 0 (black pixel) and 255 (white pixel). Therefore a straightforward normalization is obtained by dividing the whole data by 255. Two others normalization methods (computed over the data divided by 255) are implemented : the *maximum* (eq. 5) and *overall maximum* (eq. 6). Both of these can be bypassed.

$$\mathbf{F_n}(n) = \frac{\mathbf{F_p}(n)}{max(\mathbf{F_p}(n))} \tag{5}$$

$$\mathbf{F_n}(n) = \frac{\mathbf{F_p}(n)}{max_{whole}(\mathbf{F_p}(n))} \tag{6}$$

A vector containing the standard deviation of each signal is computed over the whole data fn. It is used in the threshold.

3 Synthesis

3.1 Strucutre of a single operator

The driving parameters are fed to FM operators, the output is panned with the position of the neuron on the video (position of the centroid of the bounding rectangle). A lowpass filter and a waveshaping buffer are implemented after the synthesis, but not used at the moment of writing. The neurons are equalized using an isotonic curve.

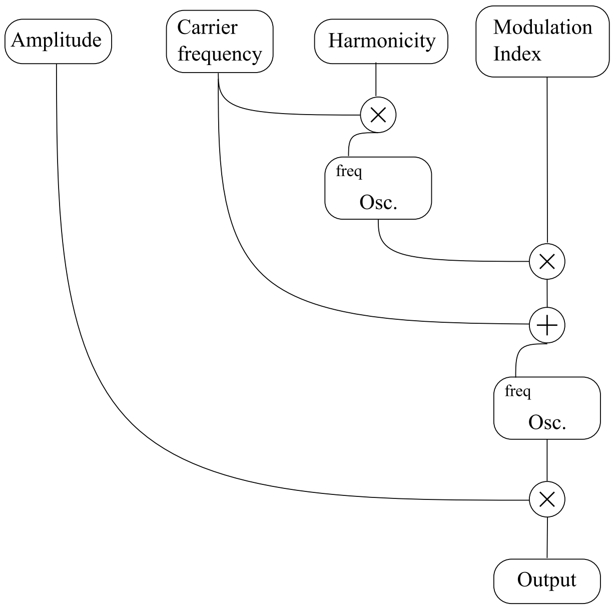

Figure 2: structure of a single operator. Both oscillator uses a sinus wavetable. Harmonicity is currently set to 1.

3.2 Group Parameter computation

Computation The group parameter is computed at each frame. In order to do so, the components of $\mathbf{F_n}(n)$ or $\frac{\mathrm{d}\mathbf{F_n}(n)}{\mathrm{d}t}$ are grouped by clusters for computation. Four group parameters are currently implemented : *Average*, *Average first derivative*, *activation ratio*, and *activation ratio first derivative*. In the following N_g designates the number of neurons in the cluster g, $\mathbf{F_n}(n)_g$ (resp. $\frac{\mathrm{d}\mathbf{F_n}(n)}{\mathrm{d}t}$) designates the signals (resp. first derivative of the signal) associated with the neurons belonging to the cluster g. For the sake of simplicity, in the following we only consider the use of the fluorescence, but the expressions are equivalent when using $\frac{\mathrm{d}\mathbf{F_n}(n)}{\mathrm{d}t}$.

- Average : $\frac{\sum_g \mathbf{F_n}(n)_g}{N_g}$, where $\sum_g$ designates the summation over the whole group g.
- Activation ratio : $\frac{\sum_g \mathbf{F_n}(n)_{g,>0}}{N_g}$ where $fn_{g,>0} = 1$ if $\mathbf{F_n}(n)_g > 0$, 0 otherwise.

The derivative of the above is computed using a backward finite difference scheme ($o(h)$).

Scaling The computed parameter is scaled using two interpolating curves. One scale the parameter itself, the other scales the modulation Index. The scaled output is amplified by a factor set by the user in order to control the range of variation.

4 Video, interface and interaction

4.1 Playback control

The playback is controlled by a counter which goes from 1 to the number of frames. The user can jump to a frame by clicking on the interface. The playing rate (i.e. the time step and the control rate) can be changed.

4.2 Overlay, neuron selection and allocation

The neuron are represented by the bounding boxes of their ROIs. Thus, the user can select them by clicking on the video. A schematic of the bound boxes is presented figure 4.2.

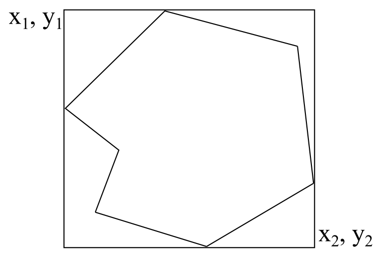

Figure 3: schematic of a bounding box and the associated ROI. The rectangle is represented by its top left and bottom right corners.

Therefore when the mouse is clicked at the point x_s, y_s, the selected neuron is so that $x_1 < x_s < x_2$ and $y2 < y_s < y_1$. When a cluster has been selected by the user, the neuron is allocated to the cluster and the overlay is updated.

The overlay displays the centroid of the bounding boxes along with the colors which represent the clusters. It is drawn at initialization and when it has to be updated.

5 implemented but not used

A waveshaping buffer is implemented. For instance it is filled with the identity. I have tried several functions (mainly x^k for different k), in order to expand the variation between low activation and high activation, but the main result was a sharper sound.

A compressor is implemented in the synthesis, but it is barely compressing, and in my opinion is not that useful.

6 Initialization sequence

The initialization process is launched in the subatch *Init*.

1 data
   - loads Clusters
   - loads Rois
     * compute the number of ROIs
   - loads raw data
   - loads notes
   - load initial gains

2 processing
   - initialize the number of neurons inside the group for the preprocessing *average*
   - Launch *average* preprocessing

3 normalization
   - Set to *bypass*
   - write the normalized data collection
   - Computes the means and standard deviation vectors

4 videoplane
   - load video
   - get number of frames, dimension of the video
   - initialize the counter with the number of frames

5 world
   - disable the window resizing by the user
   - enable the mouse tracking
   - start video rendering

6 overlay
   - get movie dimensions
   - get video dimensions
   - compute the centers of the ROIs
   - display the ROIs

7 synthesis

8 ui

9 waveshaper (buffer)

10 selected group